The Online Graph Bandwidth Problem

RAYMOND BOARD*

Division of Research and Statistics, Federal Reserve Board,
Washington, DC 20551

The online graph bandwidth problem is defined, and we present an online algorithm that always outputs a \( \frac{(2k - 1)n + 1}{2k} \)-bandwidth function for any \( n \)-vertex graph with bandwidth \( k \). A lower bound of \( \frac{k}{k+1}n-2 \) is shown for any such algorithm. Two other protocols for online problems are given, and we prove lower bounds for the bandwidth problem under both of these alternative protocols. © 1992 Academic Press, Inc.

1. INTRODUCTION

An online algorithm is an algorithm that is given a series of discrete inputs, and must make some irrevocable decision after seeing each input. An online graph algorithm is an online algorithm in which the inputs are pieces of a graph, and the decisions are usually determining what label to assign to a vertex. At least two other online graph problems have been studied in some detail. Gyárfás and Lehel (1988) and Kierstead (1988) consider online algorithms for coloring the vertices of a graph. The problem of constructing chain covers and antichain covers of partially ordered sets online has been studied in (Kierstead, 1981, 1986). Online algorithms for a variety of other problems, such as packing problems (Brown et al., 1982; Yao, 1980), dynamic storage allocation (e.g., Coffman and Leighton, 1989), and metrical task systems, including server and caching problems (Borodin et al., 1987; Chrobak et al., 1989; Manasse et al., 1988; Raghavan and Snir, 1988), have also been studied. Work done on recursively colorable infinite graphs (Bean, 1976; Carstens and Päppinghaus, 1983) is related to online graph algorithms.

The study of bandwidths originally arose in connection with matrices, but was readily recast as a problem in graph theory. The problem of

* This work was completed while the author was at the Department of Computer Science, University of Illinois at Urbana–Champaign, supported in part by NSF Grant IRI-8809570. Author’s email address: mlrab01@fed.frb.gov.

0890-5401/92 $5.00
Copyright © 1992 by Academic Press, Inc.
All rights of reproduction in any form reserved.
finding the bandwidth of a graph is to determine the smallest possible value $k$ such that there exists a bijective function $f$ from the vertex set $V$ to the set $\{1, 2, \ldots, |V|\}$ with the property that if two vertices have an edge between them then the difference of their images under $f$ is no more than $k$. The problem of determining the bandwidth of an arbitrary graph is known to be NP-complete (Papadimitriou, 1976). Graph bandwidths also arise in the study of VLSI circuit design.

We are interested in the problem of finding online algorithms that construct a function $f$ with as small a bandwidth as possible for arbitrary graphs. It is not possible to always find the minimum possible bandwidth online; thus, we try to find a function with a bandwidth not too much larger than the minimum. No restrictions are placed on the computational resources (time and space) available to the algorithms. We do not consider infinite graphs.

As in other studies of online graph algorithms (Gyárfás and Lehel, 1988; Kierstead, 1988), we evaluate online algorithms according to the worst-case ratio of their performance relative to that of the best offline algorithm. Furthermore, we compare online bandwidth algorithms based on their performance on all graphs; that is, we use as our criterion the worst-case performance (relative to the best offline result) that is achieved by the algorithm over any possible graph. Note that an online algorithm that is optimal in this sense may perform significantly worse than some other online bandwidth algorithm on a particular graph.

An application of this particular problem is as follows. Suppose we receive some data files in a sequential manner, and must write each file onto a sequential tape as it arrives. The files can be placed anywhere on the tape, but we want them positioned so as to minimize the longest distance that the tape head must travel between files when the data files are subsequently accessed. If the pattern of anticipated data accesses is such that it can be modeled by a graph, then the problem of deciding where to put each file as it arrives can be modeled by an online graph bandwidth problem.

Turner (1986) also studied approximation algorithms for the graph bandwidth problem. However, he does not consider online algorithms, and he assumes an underlying probability distribution over the possible graphs and uses an average-case performance analysis. We analyze online algorithms in terms of their worst-case performance.

The outline of this paper is as follows. We first present an online algorithm for the bandwidth problem, and demonstrate that it is close to optimal. We then define two new, more powerful protocols for online algorithms, and prove lower bounds on the bandwidth of the function constructed by any algorithm that operates according to these protocols. We conclude with some suggestions for future work.
Let $G$ be a simple finite undirected graph with vertex set $V = \{v_1, v_2, ..., v_n\}$ and edge set $E$. Note that $|V| = n$. If $(u, v) \in E$ then $u$ and $v$ are said to be adjacent. For any $v_i \in V$ we define the adjacency list for $v_i$, as $\text{Adj}(v_i) = \{u: (v_i, u) \in E\}$. We define the restricted adjacency list for $v_i$ as $\text{Adj}_i(v_i) = \text{Adj}(v_i) \cap \{v_1, v_2, ..., v_{i-1}\}$.

**Definition 2.1.** For any integer $m$, $1 \leq m \leq n - 1$, an $m$-bandwidth function for a graph $G$ is a bijective function $f: V \rightarrow \{1, 2, ..., n\}$ with the property that for any edge $(u, v)$ in $E$, $|f(u) - f(v)| \leq m$. Alternatively, a function with this property may be said to have bandwidth $m$. If a function $f$ is an $m$-bandwidth function for some $m$, then $f$ is a bandwidth function. The bandwidth of $G$ is the smallest positive integer $k$ such that there exists a $k$-bandwidth function for $G$.

The size of a graph's bandwidth gives information about how the vertices in the graph are connected. In a graph with a small bandwidth, the vertices tend to have edges only to vertices in the same part of the graph, while a graph with a large bandwidth has edges between vertices in different parts of the graph. Thus the bandwidth measures certain locality properties of the edge set. Note that if $G$ has bandwidth $k$, then no vertex in $G$ can have degree greater than $2k$. In particular, if $G$ has bandwidth 0, then there are no edges in $G$, so any bijective function from $V$ onto $1, 2, ..., n$ is a 0-bandwidth function for $G$. In this paper we will assume that the edge set $E$ is not empty, and thus $G$ has bandwidth $k \geq 1$.

The problem studied here is the construction of an $m$-bandwidth function $f$ by an algorithm $A$ when $A$ is given its inputs, and outputs the values of $f$, according to an online protocol (defined below).

**Definition 2.2.** An algorithm $A$ is an online bandwidth algorithm if its input/output behavior is as follows. Initially $A$ is given as input (for some graph $G$) the number of vertices $n$, and the bandwidth $k$. Then, for some ordering of the vertices, $v_1, v_2, ..., v_n$, $A$ is presented the restricted adjacency lists of the vertices in that order. After the list for $v_i$ is seen, $A$ must output the value of $f(v_i)$ before it is shown $\text{Adj}_{i+1}(v_{i+1})$. The decision made by $A$ as to the value of $f(v_i)$ is irrevocable. When all of the restricted adjacency lists have been seen by $A$, it must have defined the values of $f$ such that $f$ is a bandwidth function for $G$.

**Definition 2.3.** An online bandwidth algorithm $A$ is an online $m$-bandwidth algorithm if, for any $n$ and $k$, for any graph $G$ with $n$ vertices and bandwidth $k$, and for any ordering of the vertices of $G$, the function $f$ defined by $A$ is an $m$-bandwidth function.
Note that it is trivial to find an online \((n-1)\)-bandwidth algorithm. In fact, any algorithm that produces a bijective function from \(V\) onto \(\{1, 2, \ldots, n\}\) (according to the online protocol) is an online \((n-1)\)-bandwidth algorithm.

This definition of an online algorithm is generally similar to the method of presenting graphs and partially ordered sets used in other work on online graph algorithms. One difference between our definition and the protocols used in the problems of online graph coloring and recursively covering posets with chains/antichains is that we allow the algorithm to know the number of vertices in the graph. In the coloring and poset problems, the objective is to construct a function with domain \(V\) and a range as small as possible, provided that it satisfies certain constraints. In the bandwidth problem, however, the range of the function must be the same size as \(V\); thus, any online algorithm would be severely handicapped if it did not know what the range was required to be. Note that we also permit an online bandwidth algorithm to know in advance the actual bandwidth of the graph. Because of the stringent requirements on the algorithm (i.e., that it must construct a bijective function with the desired properties based on only partial information about the graph) we feel that it is not unreasonable to provide the algorithm with this information.

3. An Online Algorithm for Finding the Bandwidth of a Graph

**Theorem 3.1.** There exists an online \(((2k-1)n+1)/2k\)-bandwidth algorithm.

Note that an alternative way to phrase the problem and the above result is as follows. The definition of an online bandwidth algorithm could be changed to drop the condition that the algorithm is given the value of \(k\). Then the above theorem could state that for any \(n\) and \(k\), there is an online \(((2k-1)n+1)/2k\)-bandwidth algorithm for the set of all graphs with \(n\) vertices and bandwidth \(k\).

**Proof:** Define \(B(n, k) = (((2k-1)n+1)/2k)\). The online algorithm OLBW computes a \(B(n, k)\)-bandwidth function \(f\).

**Algorithm OLBW.**

1. Set \(\text{LABELS} = \{1, 2, \ldots, n\}\).
2. Set \(\mu = (n+1)/2\).
3. For each \(i = 1, 2, \ldots, n\) do:
(i) Define \( f(v_i) = z \), where \( z \) is the element in LABELS that maximizes \( |z - \mu| \), subject to the constraint that for each \( v_j \in \text{Adj}_i(v_i) \),

\[
|f(v_i) - f(v_j)| \leq B(n, k).
\]

In case of ties, choose the smaller value.

(ii) Set \( \text{LABELS} = \text{LABELS} - \{z\} \).

Note that the algorithm OLBW sets \( f(v_i) \) equal to the unused value in \( \{1, 2, ..., n\} \) furthest from \( \mu \) that is still consistent with an eventual online bandwidth of \( B(n, k) \). Since \( \mu = (n + 1)/2 \), \( \mu \) is in the "middle" of 1, 2, ..., \( n \).

**Definition 3.2.** Let \( \alpha \) and \( \beta \) be elements of \( \{1, 2, ..., n\} \). \( \alpha \) is more extreme than \( \beta \) (or \( \beta \) is less extreme than \( \alpha \)) if \( |\alpha - \mu| > |\beta - \mu| \). \( \alpha \) is at least as extreme as \( \beta \) (or \( \beta \) is no more extreme than \( \alpha \)) if \( |\alpha - \mu| \geq |\beta - \mu| \).

In the following we will frequently refer to the assignment of an image under \( f \) to a vertex \( v_i \) as "labeling \( v_i \)" or "giving \( v_i \) a label." Similarly, elements in \( \text{LABELS} \) will be referred to as "unused labels" or "available labels," while elements of \( \{1, 2, ..., n\} \) that are no longer in \( \text{LABELS} \) will be referred to as "used labels." Thus OLBW sets \( f(v_i) \) equal to the most extreme unused label that is consistent with \( f \) having a bandwidth of at most \( B(n, k) \).

\( f \) is well defined and bijective. To show that \( f \) has bandwidth \( B(n, k) \), we will assume otherwise and show that a contradiction inevitably arises.

Suppose that \( f \) has a bandwidth greater than \( B(n, k) \). Let \( v_s \) be the first vertex encountered by OLBW such that labeling \( v_s \) violates the bandwidth constraint; that is, while OLBW is processing \( v_s \), it finds that there is no element in \( \text{LABELS} \) that satisfies the constraint in Step 3(i) of the algorithm.

**Case 0.** \( \text{Adj}_s(v_s) = \emptyset \), i.e., there are no edges in \( E \) between \( v_s \) and any previously seen vertex. Then any label that is given to \( v_s \) fails to increase the bandwidth of \( f \). Thus the constraint of Step 3(i) cannot have been violated by \( v_s \), after all, and we get a contradiction.

**Case 1.** \( \text{Adj}_s(v_s) = \{u\} \). Thus \( v_s \) has an edge to exactly one previously seen vertex, which we will call \( u \). Since \( u \) was processed before \( v_s \), OLBW has already computed \( f(u) \).

**Fact 3.3.** For any \( k \geq 1 \), \( n - B(n, k) < B(n, k) + 1 \).

If \( n - B(n, k) \leq f(u) \leq B(n, k) + 1 \), then \( |f(v_s) - f(u)| \) is at most either \( n - B(n, k) = B(n, k) \) or \( B(n, k) + 1 - 1 = B(n, k) \). Hence no label that is given to \( v_s \) will cause the bandwidth of \( f \) to exceed \( B(n, k) \). Thus we need only consider the cases when \( f(u) < n - B(n, k) \) or \( f(u) > B(n, k) + 1 \).
Case 1-A. $f(u) < n - B(n, k)$. Let $t$ be the largest integer such that all of the labels from 1 to $t$ have already been used; thus $t + 1$ is the smallest unused label. Let $m = t + 1$. Note that if there were any unused labels between 1 and $B(n, k) + 1$, then $v_s$ could be given one of those labels. By Fact 3.3, $n - B(n, k) < B(n, k) + 1$, so $|f(v_s) - f(u)|$ would be at most $(B(n, k) + 1) - 1 = B(n, k)$, and the bandwidth of $f$ would not be forced to exceed $B(n, k)$. Thus we can assume that $t > B(n, k)$, and hence $m \geq B(n, k) + 2$.

For any used label $p$, let $f^{-1}(p)$ be the vertex that has been assigned the label $p$ by OLBW.

Consider the set $P = \{1, 2, \ldots, t\}$. All of the elements of $P$ are labels that have already been used.

**Definition 3.4.** We define the sets $P_1$ and $P_2$ as follows:

- $P_1$ is the set of labels $p$ in $P$ such that $p$ is at least as extreme as $m$.
- $P_2$ is the set of labels $p$ in $P$ such that there is an edge in $E$ from $f^{-1}(p)$ to a vertex that has already been given a label less than $m - B(n, k)$.

**Lemma 3.5.** $P_1 \cup P_2 = P$.

*Proof.* Consider any $p \in P$. Let $v_j (j < s)$ be the vertex that was assigned $p$ as its label. By Step 3(i) of the algorithm OLBW, $p$ was at that time the most extreme element in LABELS that would not, if assigned to $v_j$, force the bandwidth of $f$ to exceed $B(n, k)$. Suppose that $p \notin P_1$, so $m$ is more extreme than $p$. Then the reason that $v_j$ was given $p$, rather than $m$, as a label by OLBW must have been because assigning $m$ to $v_j$ would make the bandwidth of $f$ too large. Since $m \geq B(n, k) + 2$, the only way that this could happen would be if there was an edge from $v_j$ to a vertex that had already been assigned a label smaller than $m - B(n, k)$. Thus $p \in P_2$. 

**Lemma 3.6.** $|P_1| = n - m + 1$. $|P_2| \leq 2k(m - B(n, k) - 1)$.

*Proof.* Since $m \geq B(n, k) + 2$, $m$ is greater than $n/2$. Thus the labels that are at least as extreme as $m$ are 1, 2, ..., $n - m + 1$ and $m, m + 1, \ldots, n$. Since $m = t + 1$, the only such labels that are in $P$ are 1, 2, ..., $n - m + 1$, proving the first part of the lemma. The number of vertices that have already been assigned a label smaller than $m - B(n, k)$ is clearly bounded by $m - B(n, k) - 1$. Since $G$ has bandwidth $k$, each such vertex can have degree no more than $2k$, proving the remainder of the lemma.

Define $P_{12} = P_1 \cap \overline{P_2}$. Clearly $|P_{12}| \leq |P_1| = n - m + 1 = n - t$. Thus

$$|P_2| \geq t - |P_{12}| \geq t - (n - t) = 2t - n. \quad (1)$$
By Lemma 3.6,

$$|P_2| \leq 2k(m - B(n, k) - 1)$$

$$= 2km - ((2k - 1)n + 1) - 2k$$

$$= 2k(m - n - 1) + n - 1$$

$$= 2k(t - n) + n - 1$$

$$= 2kt - (2k - 1)n - 1.$$ 

**Lemma 3.7.** $2kt - (2k - 1)n - 1 < 2t - n.$

*Proof.* By induction on $k.$ When $k = 1,$ we get $2t - n - 1 < 2t - n.$ Assume as the inductive hypothesis that $2jt - (2j - 1)n - 1 < 2t - n$ for some fixed $j \geq 1.$ Then

$$2(j + 1)t - (2(j + 1) - 1)n - 1 = (2jt - (2j - 1)n - 1) + 2(t - n).$$

Since $t < n,$

$$2(j + 1)t - (2(j + 1) - 1)n - 1 < 2jt - (2j - 1)n - 1,$$

so by the inductive hypothesis

$$2(j + 1)t - (2(j + 1) - 1)n - 1 < 2t - n.$$ 

Thus for any $k,$ $2kt - (2k - 1)n - 1 < 2t - n.$

Thus $|P_2| < 2t - n,$ contradicting (1). Since we get a contradiction, this case cannot arise.

Case 1-B. $f(u) > B(n, k) + 1.$ Since this case is symmetric to Case 1-A, the exposition will be shorter. Define $t$ to be minimal such that all of the labels $t, t + 1, \ldots, n$ have already been used. Let $m = t - 1,$ the largest unused label. If there were any unused labels between $n - B(n, k)$ and $n,$ then $u,$ could be assigned one of them, without forcing $f$'s bandwidth to exceed $B(n, k),$ by Fact 3.3. Thus assume that $t < n - B(n, k)$ and $m < n - B(n, k) - 1.$ Define $P = \{t, t + 1, \ldots, n\}.$

**Definition 3.8.** We define the sets $P_3$ and $P_4$ as follows:

- $P_3$ is the set of labels $p$ in $P$ such that $p$ is at least as extreme as $m.$
- $P_4$ is the set of labels $p$ in $P$ such that there is an edge in $E$ from $f^{-1}(p)$ to a vertex that has already been given a label greater than $m + B(n, k).$
Lemma 3.9. $P_3 \cup P_4 = P$.

Proof. Similar to proof of Lemma 3.5. □

Lemma 3.10. $|P_3| = m$. $|P_4| \leq 2k(n - m - B(n, k))$.

Proof. Since $m \leq n - B(n, k) - 1 < n/2$, the only labels in $P$ that are at least as extreme as $m$ are $n - m + 1, n - m + 2, \ldots, n$. There are $m$ of these, proving the first equality. No more than $n - m - B(n, k)$ vertices can be assigned labels larger than $m + B(n, k)$; each such vertex has degree no more than $2k$. This proves the rest of the lemma. □

Let $P_{\bar{3}\bar{4}} = P_3 \cap P_4$. Then $|P_{\bar{3}\bar{4}}| \leq |P_3| = m = t - 1$ and

$$|P_{\bar{4}}| \geq |P| - |P_{\bar{3}\bar{4}}| = n - t + 1 - |P_{\bar{3}\bar{4}}|.$$ (2)

By Lemma 3.10,

$$|P_{\bar{4}}| \leq 2k(n - m - B(n, k))$$
$$= 2kn - 2km - (2k - 1)n - 1$$
$$= n - 2kt + 2k - 1$$
$$= (n - 2t) - (2t(k - 1) - 2k + 1).$$

Since $t \geq 2$ (because not all labels can have been used already) and $k \geq 1$,

$$2t(k - 1) - 2k + 1 \geq 4(k - 1) - 2k + 1 = 2k - 3 \geq -1.$$

Thus

$$|P_{\bar{4}}| \leq (n - 2t) - (-1) = n - 2t + 1.$$

Since $|P_{\bar{3}\bar{4}}| \leq t - 1$,

$$|P_{\bar{4}}| \leq n - t - |P_{\bar{3}\bar{4}}| < n - t + 1 - |P_{\bar{3}\bar{4}}|,$$

which contradicts (2). Hence this case cannot arise.

Case 2. $|\text{Adj}_4(v_4)| \geq 2$; $v_4$ has an edge to two or more previously seen vertices. Let $l$ and $r$ be the smallest and largest labels, respectively, of any vertex in $\text{Adj}_4(v_4)$ (if the labels $1, 2, \ldots, n$ are thought of as being written in ascending order, then $l$ is the “leftmost,” and $r$ the “rightmost,” label of any vertex in $\text{Adj}_4(v_4)$). Note that $r - l \leq n \leq 2B(n, k)$, so $r - B(n, k) \leq l + B(n, k)$. Any label between $\max\{1, r - B(n, k)\}$ and $\min\{n, l + B(n, k)\}$, inclusive, is within $B(n, k)$ of both $l$ and $r$. Thus all such labels must have
already been used since, by hypothesis, any available label that is assigned to \( v \), causes \( f \)'s bandwidth to exceed \( B(n, k) \).

We split this case into four subcases.

**Case 2-A.** \( r - B(n, k) \leq 1 \) and \( l + B(n, k) \geq n \). Thus all of the labels \( 1, 2, ..., n \) have been used already, so all of the vertices have been labeled. There is no \( v \), left to label.

**Case 2-B.** \( r - B(n, k) \leq 1 \) and \( l + B(n, k) < n \). Thus all of the labels from \( 1 \) through \( l + B(n, k) \) have been used.

Let \( t \) be maximal such that all of the labels \( 1, 2, ..., t \) have been used; thus \( t \geq B(n, k) + 1 \). If the argument in Case 1-A is repeated using \( u \in \text{Adj}_x(v_z) \), instead of \( \text{Adj}_x(v_z) = \{u\} \), then this situation is seen not to be achievable; hence this case cannot arise.

**Case 2-C.** \( r - B(n, k) > 1 \) and \( l + B(n, k) \geq n \). Thus all of the labels from \( r - B(n, k) \) through \( n \) have been used.

Let \( t \) be minimal such that all of the labels \( t, t + 1, ..., n \) have been used; thus \( t \leq r - B(n, k) \leq n - B(n, k) \). If the argument in Case 1-B is repeated with \( u \in \text{Adj}_x(v_z) \), rather than \( \text{Adj}_x(v_z) = \{u\} \), then this situation is seen to be impossible; hence this case cannot arise.

**Case 2-D.** \( r - B(n, k) > 1 \) and \( l + B(n, k) < n \). Thus all of the labels from \( r - B(n, k) \) through \( l + B(n, k) \) have been used.

Define \( a \) to be minimal, and \( b \) maximal, such that all of the labels \( a + 1, a + 2, ..., b - 2, b - 1 \) have been used already, and

\[
\{r - B(n, k), r - B(n, k) + 1, ..., l + B(n, k)\} \subseteq \{a + 1, a + 2, ..., b - 1\}.
\]

Note that \( a \) and \( b \) have not yet been used, and that \( a < r - B(n, k) \) and \( b > l + B(n, k) \). Let \( P = \{a + 1, a + 2, ..., b - 2, b - 1\} \).

**Definition 3.11.** We define the sets \( P_5, P_6, P_7, \) and \( P_8 \) as follows.

- \( P_5 \) is the set of labels \( p \) in \( P \) that satisfy both of the following conditions:
  
  1. \( p \) is more extreme than \( a \) and more extreme than \( b \).
2. \( f^{-1}(p) \) is not adjacent to any vertex that has a label either greater than \( a + B(n, k) \) or less than \( b - B(n, k) \).

- \( P_6 \) is the set of labels \( p \) in \( P \) that satisfy the following three conditions:

  1. \( p \) is more extreme than \( a \).
  2. \( f^{-1}(p) \) is not adjacent to any vertex that has been given a label greater than \( a + B(n, k) \).
  3. \( f^{-1}(p) \) is adjacent to a vertex that has been given a label less than \( b - B(n, k) \).

- \( P_7 \) is the set of labels \( p \) in \( P \) that satisfy the following three conditions:

  1. \( p \) is more extreme than \( b \).
  2. \( f^{-1}(p) \) is adjacent to a vertex that has been given a label greater than \( a + B(n, k) \).
  3. \( f^{-1}(p) \) is not adjacent to any vertex that has been given a label less than \( b - B(n, k) \).

- \( P_8 \) is the set of labels \( p \) in \( P \) that satisfy both of the following conditions:

  1. \( f^{-1}(p) \) is adjacent to a vertex that has been given a label greater than \( a + B(n, k) \).
  2. \( f^{-1}(p) \) is adjacent to a vertex that has been given a label less than \( b - B(n, k) \).

**Lemma 3.12.** \( |P| = |P_6| + |P_7| + |P_8| \).

**Proof.** Each \( p \in P \) was selected by OLBW as the label for some vertex \( f^{-1}(p) \), rather than \( a \) or \( b \). The possible reasons that \( p \) was chosen instead of \( a \) or \( b \) are as follows:

1. \( f^{-1}(p) \) is adjacent to both a vertex with a label more than \( B(n, k) \) away from \( a \) and a vertex with a label more than \( B(n, k) \) away from \( b \). Thus neither \( a \) nor \( b \) would have been chosen instead of \( p \). Note that since \( a < r - B(n, k) \leq n - B(n, k) < B(n, k) + 1 \) (by Fact 3.3) that the vertex with a label more than \( B(n, k) \) away from \( a \) must have a label greater than \( a \). Similarly, observe that \( b > l + B(n, k) \geq l + B(n, k) + 1 \), so \( n - b < \)
n - B(n, k) - 1 < B(n, k), by Fact 3.3. Hence the vertex with a label more than \( B(n, k) \) away from \( b \) must have a label less than \( b \). Any such \( p \) is contained in \( P_8 \).

2. \( f^{-1}(p) \) is adjacent to a vertex with a label more than \( B(n, k) \) away from \( a \), so \( a \) would not have been chosen. Furthermore, \( p \) is more extreme than \( b \), so \( b \) would not have been chosen. Any such \( p \) is contained in \( P_7 \cup P_8 \).

3. \( f^{-1}(p) \) is adjacent to a vertex with a label more than \( B(n, k) \) away from \( b \), so \( b \) would not have been chosen. Furthermore, \( p \) is more extreme than \( a \), so \( a \) would not have been chosen. Any such \( p \) is contained in \( P_6 \cup P_8 \).

4. The only other possible reason would be that \( p \) is more extreme than both \( a \) and \( b \). Since \( a < p < b \), this is impossible. Thus \( P_5 = \emptyset \).

Thus

\[ P \subseteq P_5 \cup P_6 \cup P_7 \cup P_8 = P_6 \cup P_7 \cup P_8. \]

Since \( P_6 \cup P_7 \cup P_8 \subseteq P \), we have \( P = P_6 \cup P_7 \cup P_8 \), and thus \( |P| = |P_6 \cup P_7 \cup P_8| \). It is immediate from their definitions that \( P_6, P_7, \) and \( P_8 \) are disjoint sets. Therefore

\[ |P| = |P_6| + |P_7| + |P_8|. \]

We now make one (final) case subdivision, this time depending on which of \( a \) and \( b \) is more extreme.

Case 2-D-I. \( b \) is at least as extreme as \( a \). Thus \( a + b \geq n + 1 \), so \( b \geq n - a + 1 \). Note that the elements of \( P_7 \cup P_8 \) are the labels in \( P \) that have been assigned to vertices with edges to vertices whose labels exceed \( a + B(n, k) \). Since the maximum degree of any vertex in \( V \) is \( 2k \), there are at most \( 2k \) distinct elements of \( P_7 \cup P_8 \) for each vertex with a label exceeding \( a + B(n, k) \). Hence

\[ |P_7| + |P_8| \leq 2k(n - a - B(n, k)). \]

The labels in \( P_6 \) are a subset of the set of labels in \( P \) that are strictly more extreme than \( a \). If \( b \geq n - a + 3 \), then the only labels in \( P \) more extreme than \( a \) are \( n - a + 2, n - a + 3, \ldots, b - 1 \). If \( b \) equals \( n - a + 2 \) or \( n - a + 1 \) (recall that \( b \) can be no smaller than this) then no labels in \( P \) are more extreme than \( a \). Thus

\[ |P_6| \leq \max\{(b - 1) - (n - a + 2) + 1, 0\} = \max\{b - n + a - 2, 0\}. \]
Since \( a + b \geq n + 1, b - n + a - 2 \geq -1 \). Thus \( b - n + a - 1 \geq 0 \), so
\[
|P_6| \leq \max\{b - n + a - 1, 0\} = b - n + a - 1.
\]

Therefore
\[
|P| = |P_6| + |P_7| + |P_8|
\leq 2k(n - a - B(n, k)) + b - n + a - 1
= 2kn - 2ka - (2k - 1)n - 1 + b - n + a - 1
= a + b - 2ka - 2
= b - a + 2a - 2ka - 2
= (b - a - 1) - (2a(k - 1) + 1)
< b - a - 1.
\]

But by the definition of \( P \) it is obvious that \( |P| = b - a - 1 \). Thus we have derived a contradiction, so this case cannot occur.

**Case 2-D-II.** \( a \) is strictly more extreme than \( b \). Thus \( a + b \leq n \), so \( a \leq n \) \( b \). The elements of \( P_6 \cup P_8 \) are the labels in \( P \) that have been assigned to vertices with edges to vertices whose labels are less than \( b - B(n, k) \). Thus
\[
|P_6| + |P_8| \leq 2k(b - B(n, k) - 1).
\]

The labels in \( P_7 \) are each labels in \( P \) that are more extreme than \( b \). If \( a \leq n - b - 1 \), then the only labels in \( P \) more extreme than \( b \) are \( a + 1, a + 2, ..., n - b \). If \( a = n - b \), then no labels in \( P \) are more extreme than \( b \). Thus
\[
|P_7| \leq \max\{(n - b) - (a + 1) + 1, 0\} = \max\{n - b - a, 0\} = n - b - a.
\]

Therefore
\[
|P| = |P_6| + |P_7| + |P_8|
\leq 2k(b - B(n, k) - 1) + n - b - a
= 2kb - (2k - 1)n - 1 - 2k + n - b - a
= (2k - 1)b - a + 2n - 2kn - 2k - 1.
\]

**Lemma 3.13.** \((2k - 1)b - a + 2n - 2kn - 2k - 1 < b - a - 1\).
Proof. Let $D = (2k-1)b - a + 2n - 2kn - 2k - 1$. Then

\[
D = D + b - b = (2k-1)b - a + 2n - 2kn - 2k - 1 + b - b = (b-a-1) + (2kb - 2b - 2kn + 2n - 2k) = (b-a-1) + ((b-n)(2k-2) - 2k).
\]

Since $b \leq n$, $b - n$ is nonpositive. Since $k \geq 1$, $2k - 2$ is nonnegative. Thus $(b-n)(2k-2)$ is nonpositive and

\[
D \leq b - a - 2k \leq b - a - 3 < b - a - 1.
\]

Since $|P| = b - a - 1$, we have derived a contradiction, so this case cannot occur.

Therefore, if we assume that $v_s$ is the first vertex that OLBW cannot assign a label to without forcing the bandwidth of $f$ to exceed $B(n, k)$, we inevitably find a contradiction. Hence no such $v_s$ can exist, and OLBW always produces a function $f$ with bandwidth no more than $B(n, k)$.

This concludes the proof of Theorem 3.1.

Corollary 3.14. The above result holds when $G$ is any graph of degree no more than $2k$.

Proof. In the proof above $G$ is assumed to have bandwidth $k$. However, the only consequence of this that is used is that $G$ must then have degree less than or equal to $2k$.

It is clear that if $k$ is large the algorithm OLBW does not guarantee an online bandwidth that is necessarily much better than the bandwidth of $n-1$ that is trivial to achieve. The result in the next section shows, however, that the performance guarantee that OLBW offers is close to optimal.

4. A Lower Bound

In this section we give a lower bound on the bandwidth of the function output by any online bandwidth algorithm.

Theorem 4.1. For any $n$ and $k$, and for any online bandwidth algorithm $A$, there exists a graph $G$ with $n$ vertices and bandwidth $k$ such that the function $f$ output by $A$ has bandwidth greater than $(k/(k+1))n - 2$). Thus no online $((k/(k+1))n - 2)$-bandwidth algorithm exists.
Proof. Given \( n, k \), and any algorithm \( A \) satisfying the hypothesis, we will define a graph \( G \) with the advertised properties.

We define \( G \) by describing the restricted adjacency lists that \( A \) is presented for each vertex. Without loss of generality, assume that \( A \) sees the restricted adjacency lists for the vertices in the order \( v_1, v_2, \ldots, v_n \).

We partition the set of labels \( \{1, 2, \ldots, n\} \) into three disjoint subsets, \( L \), \( M \), and \( R \). These are defined by

\[
L = \left\{ 1, 2, \ldots, \left\lceil \frac{n}{2k+2} \right\rceil + 1 \right\},
\]

\[
M = \left\{ \left\lfloor \frac{n}{2k+2} \right\rfloor + 2, \left\lfloor \frac{n}{2k+2} \right\rfloor + 3, \ldots, n - \left\lceil \frac{n}{2k+2} \right\rceil - 1 \right\},
\]

and

\[
R = \left\{ n - \left\lfloor \frac{n}{2k+2} \right\rfloor, n - \left\lceil \frac{n}{2k+2} \right\rceil + 1, \ldots, n \right\}.
\]

The restricted adjacency lists given as input to \( A \) are as follows. Let \( v_i \) be the vertex currently under consideration. If there are unused labels remaining in \( L \) and unused labels still in \( R \), then \( \text{Adj}_i(v_i) = \emptyset \). Otherwise, at least one of \( L \) and \( R \) has had all of its labels assigned to vertices. Define \( X \) to be the first of \( L \) and \( R \) to have all of its labels used. Let \( x \) be the most extreme label in \( X \) such that \(|\text{Adj}(f^{-1}(x)) \cap \{1, 2, \ldots, i-1\}| < 2k\). Then define \( \text{Adj}_i(v_i) = \{f^{-1}(x)\} \). If no such \( x \) exists (i.e., if each label in \( X \) is assigned to a vertex already on \( 2k \) edges), then define \( \text{Adj}_i(v_i) = \emptyset \).

To see that \( G \) has the desired properties, assume that \( X = L \) (the case of \( X = R \) is symmetric). Define \( V_L = \{v \in V : f(v) \in L\} \), \( V_M = \{v \in V : f(v) \in M\} \), and \( V_R = \{v \in V : f(v) \in R\} \). Clearly \( V_L \), \( V_M \), and \( V_R \) partition \( V \). Note that each edge in \( G \) has exactly one of its vertices in \( V_L \). We want to show that there exists an edge connecting a vertex in \( V_L \) with a vertex in \( V_R \). Consider the point at which the last remaining label in \( L \) was assigned to some vertex. At this time there was still at least one unused label in \( R \) (recall that we are assuming that \( L \) was the first of \( L \) and \( R \) to have all of its labels used). Note that if the number of possible edges incident to vertices in \( V_L \) is greater than the number of unused labels in \( M \), then the as yet unlabeled vertices in \( V_R \) will eventually be connected to vertices in \( V_L \). Thus the only way that an edge between vertices in \( V_L \) and \( V_R \) can be avoided is if the number of unused labels in \( M \) exceeds the number of possible edges incident to vertices in \( V_L \). Since \( G \) is to have bandwidth \( k \), its vertices may have degree as large as \( 2k \). Thus the number of possible edges incident to vertices in \( V_L \) is \( 2k \cdot |V_L| \geq (k/(k+1))n + 2k \). The number of unused labels in \( M \) cannot exceed \( |M| \leq (k/(k+1))n - 2 \), which is less
than the number of possible edges to vertices in $V_L$. Thus there must exist some edge between vertices in $V_L$ and $V_R$. Hence the bandwidth of $f$ is at least

$$\left( n - \left\lfloor \frac{n}{2k+2} \right\rfloor \right) \cdot \left( \left\lceil \frac{n}{2k+2} \right\rceil + 1 \right) \geq \frac{k}{k+1} n - 2.$$ 

It remains to be shown that $G$ has bandwidth $k$. Each vertex in $V_L$ has degree no more than $2k$, and each vertex in $V_M \cup V_R$ has degree either 0 or 1. Let $l_1, l_2, \ldots, l_{|L|}$ be the vertices in $V_L$. For each $i = 1, 2, \ldots, |L|$, let $u_1^i, u_2^i, \ldots, u_k^i$ be the vertices in $V_M \cup V_R$ adjacent to $l_i$. Each $k_i \leq 2k$. Finally, let $z_1, z_2, \ldots, z_Y$ be the vertices in $V_M \cup V_R$ of degree 0. We can order the vertices of $V$ as follows:

$$u_1^1, u_2^1, \ldots, u_{k_1/2}^1, l_1, u_{k_1/2+1}^1, \ldots, u_{k_1}^1,$$

$$u_1^2, u_2^2, \ldots, u_{k_2/2}^2, l_2, u_{k_2/2+1}^2, \ldots, u_{k_2}^2,$$

$$\vdots$$

$$\vdots$$

$$\vdots$$

$$u_1^{|L|}, u_2^{|L|}, \ldots, u_{k_{|L|/2}}^{|L|}, l_{|L|}, u_{k_{|L|/2}+1}^{|L|}, \ldots, u_{k_{|L|}}^{|L|}, z_1, z_2, \ldots, z_Y.$$ 

Now, define a bandwidth function $g$, such that the vertex in position $p$ of the above sequence is assigned the label $p$ by $g$. Since there are no edges between vertices in $V_L$, it is easy to see that $g$ is a $k$-bandwidth function for $G$, and hence that $G$ has bandwidth $k$.

As was mentioned above, the proof of the case that all of the labels in $R$ are used before all of the labels in $L$ is symmetric, and hence omitted. \(\blacksquare\)

Note that the difference between the result achievable by the algorithm OLBW in Theorem 3.1 and this lower bound is only about $((k-1)/(2k^2+2k))n$, which is less than $n/2k$.

5. OTHER ONLINE PROTOCOLS

We wish to consider other possible protocols for online algorithms. In the protocol defined in Section 2, which we will henceforth refer to as Protocol 1, the information that the online algorithm was given for each vertex was limited to a list of the vertices in its adjacency list that it had already labeled. We define two new online protocols, both of which permit an algorithm to see more of the graph before producing its output than is allowed under Protocol 1. We then prove lower bounds on the bandwidths
DEFINITION 5.1. An algorithm \( A \) is a Protocol 2 online bandwidth algorithm if its input/output behavior is as follows. Initially \( A \) is given as input (for some graph \( G \)) the number of vertices \( n \) and the bandwidth \( k \). Then, for some ordering of the vertices \( v_1, v_2, ..., v_n \), \( A \) is presented the adjacency lists of the vertices in that order. After the list for \( v_i \) is seen, \( A \) must output the value of \( f(v_i) \) before it is shown \( \text{Adj}(v_{i+1}) \). The decision made by \( A \) as to the value of \( f(v_i) \) is irrevocable. When all of the adjacency lists have been seen by \( A \), it must have defined the values of \( f \) such that \( f \) is a bandwidth function for \( G \).

DEFINITION 5.2. A Protocol 2 online bandwidth algorithm \( A \) is a Protocol 2 online \( m \)-bandwidth algorithm if, for any \( n \) and \( k \), for any graph \( G \) with \( n \) vertices and bandwidth \( k \), and for any ordering of the vertices of \( G \), the function \( f \) defined by \( A \) is an \( m \)-bandwidth function.

Note that this type of protocol might also be adapted to other online graph problems, such as graph coloring.

THEOREM 5.3. For any \( n \) and \( k \), and for any Protocol 2 online bandwidth algorithm \( A \), there exists a graph \( G \) with \( n \) vertices and bandwidth \( k \) such that the function \( f \) output by \( A \) has bandwidth at least \( ((k - 1)/4k)n - \frac{k}{4} \). Thus there is no Protocol 2 online \( ((k - 1)/4k)n - 2 \)-bandwidth algorithm.

Thus for large \( k \) the lower bound is only about one quarter the size of the bound obtained for Protocol 1 algorithms.

Proof. Given \( n, k \), and \( A \) satisfying the hypothesis, we define a graph \( G \) with the properties described.

We define \( G \) by describing the adjacency lists of its vertices. Let \( v_1, v_2, ..., v_n \) be the vertices of \( G \) in the order in which their adjacency lists
are shown to $A$. As in the proof of Theorem 4.1, we partition the set of labels, \{1, 2, ..., $n$\}, into three sets. Define

$$L = \left\{1, 2, ..., \left\lceil \frac{n}{2k} \right\rceil + 2\right\},$$

$$M = \left\{\left\lceil \frac{n}{2k} \right\rceil + 3, \left\lceil \frac{n}{2k} \right\rceil + 4, ..., \left\lceil \frac{2k-1}{2k} n \right\rceil - 2\right\},$$

and

$$R = \left\{\left\lceil \frac{2k-1}{2k} n \right\rceil - 1, \left\lceil \frac{2k-1}{2k} n \right\rceil, ..., n\right\}.$$

Let $s = \lfloor ((2k-1)/2k)n \rfloor$.

The adjacency lists given as input to $A$ are as follows. Let $v_i$ be the current vertex. If $A$ has not yet used any of the labels in $L$, or if $A$ has not yet used any of the labels in $R$, then $\text{Adj}(v_i) = \{v_i\}$, where $t$ is minimal such that $t \geq s$ and the number of edges seen so far that are incident to $v_i$ is less than $2k - 1$ (it will be shown below that such $t < n$ exists). The other case, in which $A$ has already used labels from both $L$ and $R$, is handled as follows. Let $u_1$ and $u_2$ be the first vertices to be assigned labels in $L$ and $R$, respectively. We must define $\text{Adj}(u_i)$ for each $i > \max\{l, r\}$. For some $a, b \geq s$, $\text{Adj}(v_i) = \{v_a\}$ and $\text{Adj}(v_i) = \{v_b\}$. Define $\text{Adj}(v_a) = \{v_n\}$, $\text{Adj}(v_b) = \{v_n\}$, and $\text{Adj}(v_i) = \{v_a, v_b\}$ (we can do this since $a, b, n$ are at least $s$, which will be shown below to be greater than $\max\{l, r\}$, and thus this will not contradict any adjacency lists defined earlier). For all $j > \max\{l, r\}$ such that $j$ is not equal to $a, b, or n$, define $\text{Adj}(v_j)$ to be consistent with the edges already seen (no new edges are added).

To see that the $G$ is well defined, we must show that each adjacency list was only defined once. First, we show that $\max\{l, r\} < s$. The largest that $l$ can be is $|M| + |R| + 1$. Similarly, $r \leq |M| + |L| + 1$. Since $|L| = |R|$, we get

$$\max\{l, r\} \leq |M| + |L| + 1 = \left\lfloor \frac{2k-1}{2k} n \right\rfloor - 1 < \left\lfloor \frac{2k-1}{2k} n \right\rfloor = s. \quad (3)$$

We must also show that $v_n$ was not put into the adjacency list of any vertex other than $v_a$ and $v_b$; i.e., we must show that $t$ is always less than $n$. Since $t$ is only defined for vertices in $\{v_1, v_2, ..., v_{\max\{l, r\}}\}$, it is sufficient to demonstrate that there are enough vertices in $\{v_{s+1}, v_{s+2}, ..., v_{n-1}\}$ to have edges to $\max\{l, r\}$ different vertices. Each vertex in $\{v_{s+1}, v_{s+2}, ..., v_{n-1}\}$ is on at most $2k - 1$ edges incident to vertices in $\{v_1, v_2, ..., v_{\max\{l, r\}}\}$, so the
number of different vertices that can have edges incident to vertices in \( \{v_s, v_{s+1}, ..., v_{n-1}\} \) is

\[
|\{v_s, v_{s+1}, ..., v_{n-1}\}| (2k - 1) = (n - s)(2k - 1) \geq \frac{2k - 1}{2k} \cdot n > \max\{l, r\},
\]

by (3). Thus \( t < n \), so \( G \) is well defined.

Note that \( (v_s, v_n, v_h, v_r) \) is a path of length four from \( v_s \) to \( v_r \).

Since \( f(v_s) - f(v_r) \geq |M| + 1 \), at least one of \( f(v_s) - f(v_h) \), \( f(v_h) - f(v_n) \), \( f(v_n) - f(v_r) \), and \( f(v_r) - f(v_s) \) must be \( (|M| + 1)/4 \) or greater. Thus the bandwidth of \( f \) is at least

\[
|M| + 1 \quad 4 = \frac{n - |L| - |R| + 1}{4} > \frac{n - 2((n/2k) + 3) + 1}{4k} = \frac{k - 1}{4k} n - \frac{5}{4}.
\]

Finally, we show that \( G \) has bandwidth \( k \). Let \( q_1, q_2, ..., q_{n-s-2} \) be the vertices in \( \{v_s, v_{s+1}, ..., v_{n-1}\} \setminus \{v_a, v_b\} \). Each such vertex has degree no more than \( 2k - 1 \). For \( h = 1, 2, ..., n - s - 2 \), define \( u^h_1, u^h_2, ..., u^h_{m_h} \) to be the vertices in \( \{v_1, v_2, ..., v_{max\{l, r\}}\} \) adjacent to \( q_h \). Each such vertex has degree 1. Note that for each \( h \), \( m_h \leq 2k - 1 \). Let \( u^1_1, u^2_2, ..., u^m_{m_1} \) and \( v_I \) be the vertices in \( \{v_1, v_2, ..., v_{max\{l, r\}}\} \) adjacent to \( q \). Each of these vertices has degree 1. Similarly, let \( u^b_1, u^b_2, ..., u^b_{m_b} \) and \( v_r \) be the vertices in \( \{v_1, v_2, ..., v_{max\{l, r\}}\} \) adjacent to \( v \). Again, each such vertex has degree 1. Note that \( m_a \) and \( m_h \) are both less than or equal to \( 2k - 2 \).

We can order the vertices of \( G \) as follows:

\[
\begin{align*}
u^1_1, & u^2_2, ..., u^m_{m_1/2}, q_1, u^1_{m_1/2} + 1, ..., u_{m_1/2}, \\
u^2_2, & u^2_3, ..., u^m_{m_2/2}, q_2, u^2_{m_2/2} + 1, ..., u^2_{m_2/2}, \\
& ... \\
u^n_{n-s-2}, & u^b_2, ..., u^b_{m_b/2} + 1, v_b, u^b_{m_b/2} + 2, ..., u^b_{m_b}, v_r, v_n, v_f, \\
& u^b_1, u^a_2, ..., u^a_{m_a/2} - 1, v_a, u^a_{m_a/2}, ..., u^a_{m_a}.
\end{align*}
\]

Define a bandwidth function \( g \), such that the vertex in position \( p \) of the above sequence is assigned the label \( p \) by \( g \). It is not difficult to see that \( g \) is a \( k \)-bandwidth function for \( G \), and hence that \( G \) has bandwidth \( k \). □

A third definition of an online protocol is to allow the algorithm to see the same information as in Protocol 2, but permit the algorithm to choose which vertex it wants to label next, rather than allow an adversary to make the decision. Clearly any Protocol 2 algorithm can be readily adapted to perform according to this protocol (Protocol 3) with no loss in its power.
Since the above proof of the Protocol 2 performance bound does not work for Protocol 3 algorithms, it is possible that there are more powerful algorithms that operate under the new protocol.

**Definition 5.4.** An algorithm $A$ is a Protocol 3 online bandwidth algorithm if its input/output behavior is as follows. Initially $A$ is given as input (for some graph $G$) the number of vertices $n$ and the bandwidth $k$. $A$ then selects a vertex $v$ and is shown $\text{Adj}(v)$. After the list for $v$ is seen, $A$ outputs the value of $f(v)$. The decision made by $A$ as to the value of $f(v)$ is irrevocable. Then $A$ selects a new vertex $v$, and the process is repeated. When all of the adjacency lists have been seen by $A$, it must have defined the values of $f$ such that $f$ is a bandwidth function for $G$.

**Definition 5.5.** A Protocol 3 online bandwidth algorithm $A$ is a Protocol 3 online $m$-bandwidth algorithm if, for any $n$ and $k$, for any graph $G$ with $n$ vertices and bandwidth $k$, and for any ordering of the vertices of $G$, the function $f$ defined by $A$ is an $m$-bandwidth function.

Like Protocols 1 and 2, this protocol can also be adapted to other graph problems.

**Theorem 5.6.** For any $k > 1$, for any $\varepsilon > 0$, and for any Protocol 3 online bandwidth algorithm $A$, there exist $n$ and a graph $G$ with $n$ vertices and bandwidth $k$ such that the function $f$ output by $A$ has bandwidth greater than $(2 - \varepsilon)k$. Thus, for any $\varepsilon > 0$, there is no Protocol 3 online $(2 - \varepsilon)k$-bandwidth algorithm.

**Proof.** Given $k$, $\varepsilon$, and $A$ satisfying the hypothesis, we will define two graphs, $G_1$ and $G_2$. $G$ will be either $G_1$ or $G_2$, depending on the label $A$ gives to the first vertex it sees. $G_1$ and $G_2$ will be shown to have the advertised properties.

Choose $n$ to be an odd integer such that $n > (\max\{4, (4/\varepsilon) + 2\})k$. Note that this implies that $2k < n - 2k$. Without loss of generality, let $v_1$ be the first vertex that $A$ selects. $A$ is shown the adjacency list $\text{Adj}(v_1) = \{v_2, v_3, ..., v_{2k+1}\}$, and must then define $f(v_1)$.

Suppose that $2k < f(v_1) < n - 2k$. We then set $G = G_1$, where $G_1$ is defined by the following adjacency lists. For $i = 2, 3, ..., 2k + 1$, let

$$\text{Adj}(v_i) = \{v_i\}.$$ 

For $i = 2k + 2, 2k + 3, ..., n$, let

$$\text{Adj}(v_i) = \{v_i-k, v_i-k+1, ..., v_{i-1}, v_{i+1}, v_{i+2}, ..., v_{i+k}\} \cap \{v_{2k+2}, v_{2k+3}, ..., v_n\}.$$
All subsequent responses to $A$ are then made according to these adjacency lists.

Note that $G_1$ consists of two connected components. The first consists of the subgraph induced by $\{v_1, v_2, \ldots, v_{2k+1}\}$. There is an edge from $v_1$ to every other vertex in this subgraph, and these are the only edges in the subgraph. Because of the nature of this component, we will refer to it as the star. The remaining vertices induce the other connected component; in this subgraph each vertex $v_j$ has an edge from every other vertex in the subgraph that has an index between $j-k$ and $j+k$, inclusive. Due to the nature of this component, we will refer to it as the $k$-chain.

To see that $G_1$ has bandwidth $k$, define $g_1$ as follows:

$$g_1(v_i) = \begin{cases} 
  k + 1 & \text{if } i = 1 \\
  i - 1 & \text{if } 2 < i \leq k + 1 \\
  i & \text{if } i \geq k + 2.
\end{cases}$$

$g_1$ is a $k$-bandwidth function for $G_1$.

Suppose that $f(v_1) < 2k$ or $f(v_1) > n - 2k$. We then set $G = G_2$, where $G_2$ is defined as follows. Order the vertices according to the sequence (recall that $n$ is odd)

$$v_n, v_{n-2}, v_{n-4}, \ldots, v_5, v_3, v_1, v_2, v_4, \ldots, v_{n-3}, v_{n-1}.$$  

There is an edge in $G_2$ between every pair of vertices that are within $k$ positions of each other in this sequence. Note that $G_2$ has bandwidth $k$, since we can define a $k$-bandwidth function $g_2$ by setting $g_2(v_i)$ equal to $v_i$'s position in the above sequence. All responses to $A$ are made according to this definition of $G_2$. Note that $\text{Adj}(v_1)$ as defined earlier is consistent with $G_2$.

It remains to be shown that the graphs $G_1$ and $G_2$ force $f$ to have a bandwidth greater than $(2 - \epsilon)k$.

**Case 1.** $2k < f(v_1) < n - 2k$, so $G = G_1$. Once again, we partition the set of labels $\{1, 2, \ldots, n\}$ into three subsets. Let $M$ be the set containing the smallest continuous sequence of labels that includes each of $f(v_1), f(v_2), \ldots, f(v_{2k+1})$. Define $L$ to be the set of labels less than the smallest label in $M$, and $R$ to be the set of labels greater than the largest label in $M$. Thus if $\text{little} = \min\{f(v_1), f(v_2), \ldots, f(v_{2k+1})\}$ and $\text{big} = \max\{f(v_1), f(v_2), \ldots, f(v_{2k+1})\}$, then $L = \{1, 2, \ldots, \text{little} - 1\}$, $M = \{\text{little}, \text{little} + 1, \ldots, \text{big}\}$, and $R = \{\text{big} + 1, \text{big} + 2, \ldots, n\}$.

**Case 1-A.** $L = R = \emptyset$. By the definition of $M$ there exist some $v_i$ and $v_j$, with $i$ and $j$ less than or equal to $2k + 1$, such that $f(v_i) = 1$ and $f(v_j) = n$. Each of $v_i$ and $v_j$ is either $v_1$ or is adjacent to $v_1$. Thus there is a path of
length two or less from $v_i$ to $v_j$. One of $f(v_j) - f(v_i)$ and $f(v_i) - f(v_j)$ must be at least $(n-1)/2$, so the bandwidth of $f$ is at least $(n-1)/2 \geq 2k$.

**Case 1-B.** Exactly one of $L$ and $R$ is equal to $\emptyset$. Assume that $L = \emptyset$ and $R \neq \emptyset$ (the proof of the other case is analogous). By the definitions of $M$ and $G_1$, $f^{-1}(1)$ is adjacent to $v_1$. Since $f(v_1) > 2k$, the bandwidth of $f$ is at least $2k$.

**Case 1-C.** $L \neq \emptyset$ and $R \neq \emptyset$. If there exist vertices $x$ and $y$ with $f(x) \in L$ and $f(y) \in R$ and such that there is an edge $(x, y)$, then the bandwidth of $f$ is at least $|M| + 1 \geq 2k + 2$. Assume no such vertices exist.

Define $V_L$, $V_M$, and $V_R$ to be the sets of vertices with labels in $L$, $M$, and $R$, respectively.

**Lemma 5.7.** There are at least $k$ vertices not in $V_L$ that are adjacent to vertices in $V_L$. There are at least $k$ vertices not in $V_R$ that are adjacent to vertices in $V_R$.

**Proof.** We prove the result for $V_L$; the proof for $V_R$ is similar. By the definitions of $L$ and $R$, all of the vertices in $V_L$ and $V_R$ are in the $k$-chain. For $i = 1, 2, \ldots, |L|$, define $v_i$, to be the $i$th-lowest indexed vertex in $V_L$; thus $l_1 < l_2 < \cdots < l_{|L|}$. Similarly, let $v_{r_i}$ be the $j$th-lowest indexed vertex in $V_R$, for $j = 1, 2, \ldots, |R|$; hence $r_1 < r_2 < \cdots < r_{|R|}$. Think of the $k$-chain as being a chain of vertices with $v_{2k+1}$ on the left end of the chain and $v_n$ on the right end, with every vertex having an edge to each vertex within distance $k$ of it. We will find a lower bound on the total number of distinct vertices in the $k$-chain that have edges to vertices in $V_L$. There are four cases to consider.

1. Suppose that $l_1 < r_1$ and $r_{|R|} > l_{|L|}$. Thus $v_{l_1}$ and $v_{r_{|R|}}$ are the leftmost and rightmost vertices, respectively, in the $k$-chain that are in $V_L \cup V_R$. Note that by the assumption above there are no edges between vertices in $V_L$ and $V_R$, so $l_{|L|} \leq n - k - 1$. The vertex $v_{l_1}$ is adjacent to at least $k$ vertices. If $l_1 \leq l_1 + k$, then $v_{l_1}$ is adjacent to at least two vertices that are not adjacent to $v_{l_1}$: $v_{l_1}$ itself and $v_{l_1+k}$. If $l_2 > l_1 + k$, then $v_{l_2}$ is adjacent to at least $k$ vertices that are not adjacent to $v_{l_1}$: each of $v_{l_2+1}$, $v_{l_2+2}$, ..., $v_{l_2+k}$. For each $i = 3, 4, \ldots, |L|$ the vertex $v_{l_i}$ is adjacent to at least one vertex ($v_{l_i+k}$) that none of $v_{l_1}, v_{l_2}, \ldots, v_{l_{i-1}}$ is adjacent to (since $l_{|L|} \leq n - k - 1$ we do not encounter the problem of running into the vertices at the right end of the $k$-chain that have degree less than $2k$). Since by hypothesis $k \geq 2$, there are at least $k + 2 + |L| - 2 = |L| + k$ vertices adjacent to vertices in $V_L$.

2. Suppose that $r_1 < l_1$ and $l_{|L|} > r_{|R|}$. Now $v_{r_1}$ and $v_{l_{|L|}}$ are the
leftmost and rightmost vertices in the $k$-chain that are in $V_L \cup V_R$. We prove the same bound as in Part 1 by an analogous proof.

Since there are no edges between vertices in $V_L$ and $V_R$, note that $l_1 \geq k + 1$. The vertex $v_{l_1}$ is adjacent to at least $k$ vertices. If $l_{|L|-1} \geq l_{|L|}-k$, then $v_{l_{|L|-1}}$ is adjacent to at least two vertices that are not adjacent to $v_{l_{|L|}}$: $v_{l_{|L|}}$ itself and $v_{l_{|L|-1}}$. If $l_{|L|-1} < l_{|L|} + k$, then $v_{l_{|L|-1}}$ is adjacent to at least $k$ vertices that are not adjacent to $v_{l_{|L|}}$: each of $v_{l_{|L|-1}}$, $v_{l_{|L|-1}-2}$, ..., $v_{l_{|L|-1}-k}$. For each $i = |L| - 2, |L| - 3, ..., 1$ the vertex $v_i$ is adjacent to at least one vertex ($v_{i-k}$) that none of $v_{l_{|L|}}, v_{l_{|L|-1}}, ..., v_{l_{i+1}}$ is adjacent to (since $l_1 \geq k + 1$ we do not encounter the problem of running into the vertices at the left end of the $k$-chain that have degree less than $2k$). Thus there are at least $k+2+|L|-2=|L|+k$ vertices adjacent to vertices in $V_L$.

3. Suppose that $l_1 < r_1$ and $l_{|L|} > r_{|R|}$, so both the leftmost and rightmost vertices in $V_L \cup V_R$ are in $V_L$. Let $r$ be the rightmost vertex in $V_R$, and let $V_{L_1}$ and $V_{L_2}$ be the sets of vertices in $V_L$ to the left and right, respectively, of $r$. An argument similar to the one given in Part 1 above shows that there are at least $|V_{L_1}| + k$ vertices adjacent to vertices in $V_{L_1}$.

Since the vertex $r \in V_R$ lies between the vertices of $V_{L_1}$ and $V_{L_2}$, and since there are no edges between vertices in $V_L$ and $V_R$, there are no vertices adjacent to vertices in both $V_{L_1}$ and $V_{L_2}$. An argument similar to the one given in Part 2 above shows that there are at least $|V_{L_2}| + k$ vertices adjacent to vertices in $V_{L_2}$. Thus there are at least $|V_{L_1}| + k + |V_{L_2}| + k = |L| + 2k$ vertices adjacent to vertices in $V_L$.

4. Suppose that $r_1 < l_1$ and $r_{|R|} > l_{|L|}$. Thus the leftmost and rightmost vertices in $V_L \cup V_R$ are in $V_R$. There are no edges between vertices in $V_L$ and $V_R$, so $l_1 \geq k + 1$ and $l_{|L|} \leq n-k-1$. The vertex $v_{l_1}$ is adjacent to at least $2k$ vertices. For each $i = 2, 3, ..., |L|$ the vertex $v_{l_i}$ is adjacent to at least one vertex ($v_{i+k}$) that none of $v_{l_1}, v_{l_2}, ..., v_{l_{i-1}}$ is adjacent to (since $l_{|L|} \leq n-k-1$ we do not encounter the problem of running into the vertices at the right end of the $k$-chain that have degree less than $2k$). Thus there are at least $2k + |L| - 1$ vertices adjacent to vertices in $V_L$.

Thus there are always at least $|L| + k$ (distinct) vertices that are adjacent to vertices in $V_L$. At least $k$ of these are not in $V_L$, proving the lemma. A similar argument shows the same result for $V_R$:
\( V_L \) and \( V_R \). Then the size of \( V_M \), and hence \( M \), is at least \( 4k + 1 \). But by the definition of \( M \), if \( m_1 \) and \( m_2 \) are the vertices in \( V_M \) with the smallest and largest, respectively, labels from \( M \), then both \( m_1 \) and \( m_2 \) are in the star. Thus there is a path of length two or less from \( m_1 \) to \( m_2 \), so \( f(m_2) - f(m_1) \geq 4k/2 = 2k \). Thus the bandwidth of \( f \) is at least \( 2k \).

Alternatively, suppose that there are \( h > 0 \) vertices in \( V_M \) that are adjacent to vertices in both \( V_L \) and \( V_R \) ("shared" vertices). Adjusting for the shared vertices, we get

\[ |V_M| \geq 2k + 1 + 2k - h = 4k + 1 - h. \]

But at least one of the \( h \) shared vertices must have a label at least \((h - 1)/2\) away from the average value of the labels in \( M \). Thus this vertex has a label at least \((|M| + 1)/2 + (h - 1)/2\) away from the label of some vertex in \( V_L \cup V_R \). Hence the bandwidth of \( f \) is at least

\[ \frac{|M| + 1}{2} + \frac{h - 1}{2} = \frac{4k + 1 - h + 1}{2} + \frac{h - 1}{2} = 2k + \frac{1}{2}. \]

**Case 2.** \( f(v_1) \leq 2k \) or \( f(v_1) \geq n - 2k \), so \( G = G_2 \). We demonstrate a lower bound on the bandwidth of \( f \) for the case when \( f(v_1) \leq 2k \). The other case is symmetric. Since \( n > ((4/\varepsilon) + 2)k \), \( 2/\varepsilon < (n/2k) - 1 \). Choose an integer \( d \) such that \( 2/\varepsilon < d < n/2k \). There are \( 2dk \) vertices with path length \( d \) or less from \( v_1 \) (not including \( v_1 \) itself). Thus at least one such vertex \( u \) must have a label of \( 2dk \) or greater. Since \( f(v_1) \leq 2k \), there is a path of length \( d \) or less from \( v_1 \) to \( u \), and \( f(u) - f(v_1) \geq 2dk - 2k \). Thus the bandwidth of \( f \) is at least

\[ \frac{2dk - 2k}{d} = 2k - \frac{2k}{d} = \left( 2 - \frac{2}{d} \right) k > (2 - \varepsilon)k. \]

This concludes the proof of Theorem 5.6.

6. Conclusion

We have defined three online protocols for constructing the bandwidth function of a graph, and have proved performance bounds for each. There are several areas ripe for future research. The performance bounds for all three protocols could be tightened. In particular, the best algorithm known under Protocols 2 and 3 is OLDB. It seems likely that there are more powerful algorithms that are specifically designed to exploit the additional information that is available under these protocols. Also, the algorithm OLDB requires only modest computational resources. Algorithms that
take better advantage of the unlimited time and space permitted by all three of these protocols might yield better results. It would also be desirable to find good algorithms that do not need to know the actual graph bandwidth at the outset.

ACKNOWLEDGMENT

I thank Lenny Pitt for his careful proofreading and helpful comments.

RECEIVED June 16, 1989; FINAL MANUSCRIPT RECEIVED November 20, 1990

REFERENCES